Scientific Research Publishing

Open Journal of Applied Sciences, 2022, 12, 723-743
https://www.scirp.org/journal/ojapps
ISSN Online: 2165-3925
ISSN Print: 2165-3917

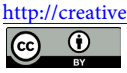
Comparison of Deep Learning Architectures for Late Blight and Early Blight Disease Detection on Potatoes

Soumo Emmanuel Arnaud[1*], Ndeda Rehema[1], Shohei Aoki[1], Murungi Lucy Kananu[2]

[1]Department of Mechatronics Engineering, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya
[2]Department of Horticulture, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya
Email: *emmanuel.soumo@gmail.com

How to cite this paper: Arnaud, S.E., Rehema, N., Aoki, S. and Kananu, M.L. (2022) Comparison of Deep Learning Architectures for Late Blight and Early Blight Disease Detection on Potatoes. *Open Journal of Applied Sciences*, **12**, 723-743.
https://doi.org/10.4236/ojapps.2022.125049

Received: March 18, 2022
Accepted: May 16, 2022
Published: May 19, 2022



Abstract

Potato late blight and early blight are common hazards to the long-term production of potatoes, impacting many farmers around the world, particularly in Africa. Early detection and treatment of the potato blight disease are critical for promoting healthy potato plant growth and ensuring adequate supply and food security for the fast-growing population. As a result, machine-driven disease detection systems may be able to overcome the constraints of traditional leaf disease diagnosis procedures, which are generally time-consuming, inaccurate, and costly. Convolutional Neural Networks (CNNs) have been shown to be effective in a variety of agricultural applications. CNNs have been shown to be helpful in detecting disease in plants because of their capacity to analyze vast volumes of data quickly and reliably. However, the method hasn't been widely used in the detection of potato late blight and early blight diseases, which reduce yields significantly. The goal of this study was to compare six cutting-edge CNN architectural models, taking into account transfer learning for training and four hyperparameters. The CNN architectures evaluated were AlexNet, GoogleNet, SqueezeNet, DenseNet121, EfficientNet b7, and VGG19. Likewise, the hyperparameters analyzed were the number of epochs, the batch size, the optimizer, and the learning rate. An open-source dataset containing 4082 images was used. The DenseNet121 architecture with a batch of 32 and a Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.01 produced the best performance, with an accuracy of 98.34% and a 97.37% f1-score. The DenseNet121 model was shown to be useful in developing computer vision systems that aid farmers in improving their disease management systems for potato cultivation.

Keywords

Image Classification, Convolutional Neural Networks, Transfer Learning,

Hyperparameters

1. Introduction

Contemporary society is concerned about food security issues due to the continual increase in population, rural to urban migration, climate change, and the reduction of cultivable land caused by the increase in industrialization and urbanization processes. The agricultural sector remains important to the socio-economic development of Africa, contributing 32% of the GDP. About 80% of the agricultural output comes from smallholder farmers and employs nearly 65% of the population. Low productivity, which characterizes agricultural production, remains a major concern in many African countries [1].

Potato is the third most important food crop in terms of global consumption [2]. The improvement of the potato production system in sub-Saharan Africa can be a pathway out of poverty. Potato has a short cropping cycle and produces a large amount per unit area in a short period (International Potato Centre, Sub-Saharan Africa 2020). However, diseases such as early blight, late blight (LB), bacterial wilt (BW), and viruses reduce the production of smallholder potato farmers in sub-Saharan Africa. Potato late blight and early blight influence the quality and quantity of the potatoes, hence causing direct crop loss. They are leaf spot diseases caused by Phytophthora infestans and the fungus Alternaria solani respectively that cause average yield losses of between 30% and 75% [3]. Typically, small-scale farmers continuously use fungicides to combat these diseases, but this practice creates a dependency on pesticides and compromises human health and the environment [4]. Furthermore, regulators such as the European Union (EU) are enacting increasingly stringent chemical usage requirements for agricultural products entering their markets [5].

Early diagnosis of plant diseases plays an important role in improving agricultural yield. Disease-infected plants typically have visible markings or lesions on their leaves, stems, flowers, and/or fruits. In general, each illness state has a distinct visual pattern that can be utilized to diagnose the disease. Small rounds or irregular dark-brown to black dots on the older (lower) leaves are the first signs of early blight (**Figure 2**). These patches can grow up to 3/8 inch in diameter and become angular-shaped over time [6]. Small, light to dark green, round to irregularly shaped water-soaked dots are the earliest signs of late blight in the field (**Figure 2**). The lowest leaves are frequently the first to get these diseases [7]. Plant diseases have traditionally been diagnosed by human experts. This is, however, costly, time-consuming, and, in some situations, unworkable; therefore, farmers are not able to respond quickly and accurately [8]. This has prompted studies that utilize deep learning, particularly used in image processing, for the early detection and management of diseases in agriculture [9]. The main contribution of this research was to determine the Convolutional Neural Network (CNN) architecture and hyperparameters that may be suitable to be deployed in

conventional as well as mobile/embedded computing environments for disease detection on potatoes in the field. To achieve it, transfer learning and fine-tuning were applied to six (6) state of art Convolutional Neural Networks (AlexNet, EfficientNet b7, GoogleNet, SqueezeNet, DenseNet121, and VGG19) to identify the hyperparameters that best influenced the training of architectures for late blight and early disease identification on potato leaves. The hyperparameters analyzed were the number of epochs, batch size, optimizer, and learning rate.

The rest of the paper is structured as follows: Section 2 provides an overview of related studies. The tests and six state-of-the-art CNN architectures are introduced in Section 3. The results are presented in Section 4. Section 5 is where the discussion takes place, and Section 6 is where the conclusions are reported.

2. Related Works

Convolutional Neural Network (CNN) is a type of artificial neural network used to interpret visual imagery in deep learning. CNNs are used in a variety of agricultural applications, including crop type classification, and the detection of diseases on plant leaves [10] [11]. The use of CNN for the detection of disease has been tested in several studies. KC *et al.* [12] used different CNN architectures to detect 58 diseases of 25 types of plants with a success rate of 99.5%. Fuentes *et al.* [13] compared the ability of several CNN architectures were able to recognize nine types of diseases and pests in tomato plants. Mohanty *et al.* [14] used the Plant Village dataset to develop a deep CNN to identify 14 crop species and 26 diseases with 38 different classes.

In the case of potatoes, studies have employed CNNs for the detection and categorization of disease. Islam *et al.* [15] created a classifier that could classify healthy leaves and those affected by late and early blight diseases using 300 potato images drawn from the PlantVillage dataset [16]. Multiclass SVM was used to identify leaf pictures into three groups based on ten color and textural attributes. The system's cross-validated accuracy was 93.7 percent. The original dataset, on the other hand, included 152 photographs of healthy potato leaves and 1000 photographs of late and early blight, respectively. Showing the system's performance on a wider dataset would have been beneficial. Sanjeev *et al.* 2021 [17] used the Feed Forward Neural Network (FFNN) Model to forecast and classify illness in potato leaves. The model's accuracy was determined to be 96.5 percent. However, FFNN needs a long training time and a large number of pairs of input and targets for the training process [18]. Lee *et al.* 2020 [19] built a CNN model for detecting early and late blight illnesses on potatoes and compared it to VGG16 and VGG19. The PlantVillage dataset, which is unbalanced, was also used by the researchers. Benchmarking these models using the evaluation metric because the dataset is imbalanced would have been beneficial.

Generally, many researchers use accuracy as a criterion for selecting the best CNNs. However, the accuracy has various flaws, including reduced uniqueness, discriminability, informativeness, and bias towards data from the majority class

[20]. Tiwari *et al.* 2020 [21] fine-tuned (transfer learning) pre-trained models like VGG19 for late blight and early blight disease detection on potato leaves. The model was fine-tuned to extract relevant characteristics from the dataset. The results were then analyzed using multiple classifiers, logistic regression outperformed the others by a significant margin of classification accuracy, achieving 97.8% on the test dataset. This study didn't describe the choice of independent variable which is one of the challenges in logistic regression. There are limited numbers of independent variables and wrongly identifying them generates errors in the model [22]. Oppenheim *et al.* 2017 [23] used CNNs for disease detection of four disease classes and an uninfected class on potatoes tuber of different shapes and sizes. The acquisition device and conditions of the 2465 pictures categorized by the trained CNN model varied. The results show that the classification method was robust enough to handle uncontrolled acquisition situations. Afzaal *et al.* [24] trained and compared 3 CNN models namely GoogleNet, VGGNet, and EfficientNet for accurate identification of early blight disease on potatoes at different growth stages using the PyTorch framework. The results showed that EfficientNet was the most effective model. There are few studies presenting models for potato late blight and early disease identification. This study presents one study on late blight and early blight with a wide dataset.

3. Materials and Methods

Figure 1 depicts a comprehensive disease dedication technique on potato leaves utilizing transfer learning. First and foremost, the data was gathered through an internet database.

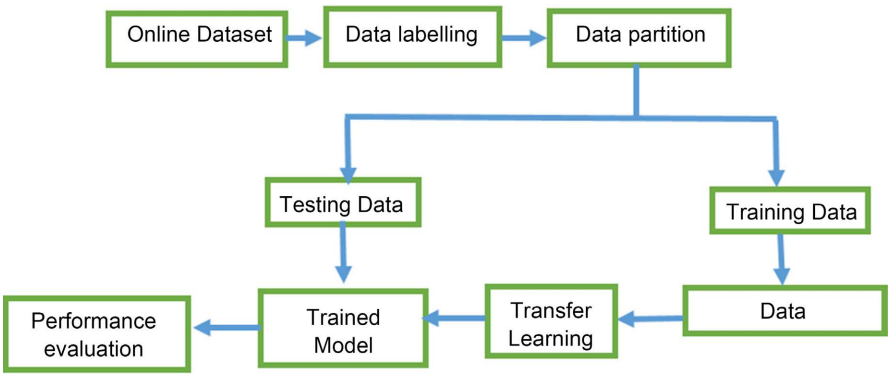


Figure 1. Overview architecture of the proposed study.

3.1. Dataset Description

The dataset was downloaded from [25] and accessed on 12 November 2021. The dataset contained 4082 images of potato leaves. The data had a spread of 3 class labels assigned to them: Healthy class, Late Blight, and Early blight. Figure 2 shows a sample of the three classes of potato images obtained from the Kaggle. Low-quality JPG images were used because of their capacity to represent real-world scenarios such as the presence of noise, contrast, and blur [17].

(A) (B) (C)

Figure 2. Example of dataset images. (A) Early blight, (B) Late blight, and (C) Healthy.

The dataset was distributed into training, validation, and testing. The training dataset was used to train the various CNN models, while the validation and test datasets were utilized for testing and evaluation of statistical measures after the CNNs were trained. As a result, the training, validation, and testing datasets were split into 80%, 10%, and 10%, respectively (**Table 1**). Different data augmentation techniques, including resizing, rotation, shear range, zoom range, horizontal flip, brightness, width shift, and height shift, were used on the training set to expand the dataset's diversity. It would solve the overfitting problem, allowing models to be more generalized. The accuracy of each CNN architecture was assessed using statistical measures including accuracy, recall, precision, specificity, and FScore.

3.2. Convolutional Neural Networks

CNNs are built by superimposing convolutional layers that apply a set of local filters through the dimensions of the data entry. Thus, these networks detect patterns by calculating local correlations using a kernel whose parameters are determined by the learning process. Kernels are tiny sections of the convolution that glide over the convolution; to extract meaningful information with fewer dimensions. The output of the preceding layer, called the feature map, is often switched to a pooling layer that reduces the dimensions of this map by sub-sampling it, thus giving the result the property of translation invariance. The reduction of the dimension is generally achieved by taking the maximum or the average of values over a set of pixels. The stacking of convolution and pooling layers at various scales can detect larger and more complex patterns [26]. These mechanisms are summarized in **Figure 3**.

It is time-consuming to collect images belonging to a specific area of interest and train a classifier from scratch. Transfer learning makes it possible to overcome this challenge by using a pre-trained model and changing its last few layers. This helps to achieve good results even with a small dataset since the basic image features have already been learned in the pre-trained model from a much larger dataset. In this study, AlexNet, GoogleNet, SqueezeNet, EfficientNetb7, VGG19, and DenseNet121 were used for transfer learning since they have shown high accuracy in earlier datasets.

Table 1. Training, validation, and testing data sets for potato Late Blight disease identification.

Label	Training/Validation/testing	Images used	Total Images
Early Blight	Training	1303	1628
	Validation	163	
	Testing	162	
Late Blight	Training	1132	1434
	Validation	161	
	Testing	141	
Healthy	Training	816	1020
	Validation	102	
	Testing	102	

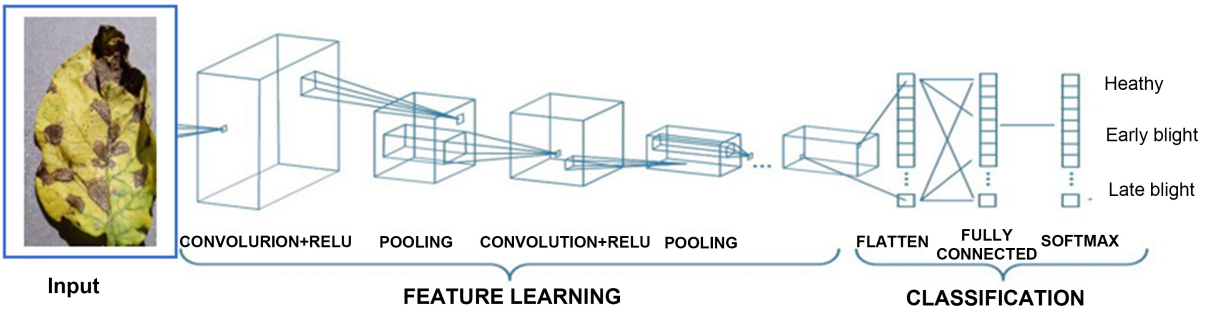


Figure 3. Overall mechanism of the CNN architecture.

3.2.1. AlexNet Architectures

The AlexNet architecture [27] consists of 5 convolution layers and 3 fully connected layers. The first convolutional layer consists of 96 filters of sizes 11 × 11 and uses a step of 4 pixels. The second layer comprises 256 filters of 5 × 5 sizes. Layers three and four use 384 filters each of size 3 × 3. The last convolutional layer has 256 filters of size 3 × 3. Finally, the 2 fully connected layers have 4096 and 1000 neurons respectively. The number of neurons in the final layer is often modified to suit the problem.

3.2.2. GoogleNet Architectures

GoogleNet is a 22-layer deep convolutional neural network. GoogleNet was built by Google to improve the performance of deep neural networks both in terms of speed and precision. Its evolution and iterative improvement led to the emergence of several versions of the network. Inception-v1 [28], Inception-v2 [29], Inception-v3 [29], and Inception-v4 [30] are the most popular versions of GoogleNet. The Inception architecture consists of 4 branches concatenated in parallel. The first branch consists of a 1 × 1 kernel convolution, followed by two 3 × 3 convolutions. The second branch comprises a 1 × 1 convolution, followed by a 3 × 3 convolution. The third branch has a pooling, followed by a 1 × 1 convolution. Finally, the last branch is a 1 × 1 convolution. Inception V4 is a result

of the combination of ResNet and Inception V3. This combination is due to the fact that, in the ImageNet Large Scale Visual Recognition Challenge 2015, ResNet and Inception V3 had nearly identical results. The two structures were combined to cut down on computational costs and improve performance. For this study, Inception V4 was used.

3.2.3. SqueezeNet

SqueezeNet architecture is a smaller network that was created to be a more compact alternative to AlexNet [31]. It consists of 26 convolutional layers and SqueezeNet. It has over 50 times fewer parameters than AlexNet but performs three times faster. SqueezeNet takes advantage of the convolution layer (which has only 1×1 filters), feeding into an expanded layer that has a mix of 1×1 and 3×3 convolution filters and chains a bunch of these modules together to arrive at a smaller model.

3.2.4. EfficientNet-b7

EfficientNet is a convolutional neural network architecture and scaling method that uniformly scales the existing model network width, depth, and resolution with a set of fixed scaling coefficients to improve its performance [32]. This is unlike conventional practice that arbitrarily scales these factors.

3.2.5. VGG19

The VGG19 model is a variation of the VGG model that has 19 layers in total (16 convolution layers, 3 fully connected layers, 5 MaxPool layers, and 1 SoftMax layer). There are other VGG variations such as VGG11, VGG16, and more. VGG architecture comprises six main structures, each of which mainly consists of multiple connected Convolutional layers and full-connected layers. The convolutional kernel is the size of 3×3, and the input size is $224 \times 224 \times 3$ (height, width, and channels). The number of layers is generally concentrated at 16-19 [33].

3.2.6. DenseNet121

Recent research has demonstrated that CNNs with shorter connections between layers adjacent to the input and those close to the output can be significantly deeper, more accurate, and more efficient to train. In the feed-forward approach, DenseNet connects each layer to every other layer. DenseNet's network has $L(L + 1)/2$ direct connections, whereas standard CNNs with L layers have L connections, one between each layer and its succeeding layer. All previous layers' feature maps are utilized as inputs into each layer, and their feature maps are used as inputs into all subsequent layers. In comparison to standard CNN, DenseNet requires fewer parameters and allows feature reuse, resulting in more compact models and better outcomes across competing datasets [34].

3.3. Hyperparameters

Optimization of a model is the process of changing the hyperparameters of the

model and observing its outputs. These hyperparameters are among others: the learning rate, epochs, optimizer, batch size, number of layers, and activation functions. In this study, the values of the optimizer, learning rate, batch size, and epochs were changed. For the optimizer, the network was trained with Adaptive Moment Estimation (Adam) and Stochastic Gradient Descent (SGD) which have good performance for image classification [27]. The learning rate for the optimizers was 0.01 and 0.001. The batch size value was 16 and 32. The number of epochs was 25 and 50. The number of layers was dependent on the CNN model.

3.4. Experimental Framework

It takes a lot of computing power to train modern CNN systems. Therefore, Google Colab GPU resources were used including the library sci-kit-learn, PyTorch, and OpenCV to perform transfer learning and evaluation of the different CNN architectures. For the standard evaluation of a classifier, several performance measures are defined. The most widely used quality metric is classification accuracy. When the test dataset contains an equal number of samples from each class, classification accuracy is a useful way to quantify performance. However, the dataset used to solve the categorization problem in question is uneven. This needs a more thorough assessment of the proposed system using additional performance indicators. The confusion matrices were utilized to investigate the tumor classification system's performance. The confusion matrix was used to describe the performance of the CNNs due to its ability to accurately measure the performance of a model with two or more classes. The Matrix checks how often its predictions are accurate compared to reality in classification problems. Each row in the matrix corresponds to a predicted class and each column corresponds to an actual class. The confusion matrix makes predictions for each row of a test dataset. Based on its predictions and the expected results, the matrix indicates the number of correct and incorrect predictions for each class. This will allow the evaluation metrics of sensitivity, accuracy, specificity, precision, and f1-score to be calculated.

The sensitivity, also known as the recall, is the proportion of potato leaves that were accurately labeled as positive to those that were truly positive. Equation (1) was used to compute the sensitivity, where TP stands for true positives, which is the number of positive cases that are correctly identified, and FN stands for false negatives, which is the number of positive cases that are incorrectly classified as negative.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{1}$$

Equation (2) represents accuracy, which estimates the percentage of samples that are correctly classified, where TN is the number of true negatives or negative cases that are negative and classified as negative, and FP is the number of false positives, which are defined as negative instances that are incorrectly classified as positive cases.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

Specificity is defined as the conditional probability of true negatives given a secondary class, which approximates the probability of the negative label being true; it is represented by Equation (3).

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{3}$$

Precision, defined as the number of images that were correctly classified as positive out of all positives, is given by Equation (4). It assesses the algorithm's prediction ability. Precision refers to how "accurate" the model is in terms of how many of the anticipated positives are actually positive.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4}$$

As illustrated in Equation (5), the F-score is calculated as the harmonic mean precision and recall. It concentrates on the study of positive classes. This statistic has a high value if the model performs well in the positive class.

$$\text{f1-score} = \frac{2 * TP}{2 * TP + FP + FN} \tag{5}$$

4. Results

This section compares six state-of-the-art CNN architectures in order to determine the best model for detecting late blight and early blight diseases in potatoes. The goal of this study was to compare CNN models based on their accuracy, precision, sensitivity, specificity, and F1-Score. The following quality indicators were used while presenting the results and comparing the models: Accuracy is defined as the ratio of correctly labeled images to the total number of samples and the F1-score (5) is a useful performance statistic, especially when there is an unequal distribution of classes. The dataset utilized in this work contained 1628 images of early blight, 1434 images of late blight, and 1020 healthy leaves. As a result, the model/optimizer with the highest f1-score was deemed the most suited architecture for the detection of potato late blight and early blight diseases in the field.

Validation accuracy ranged from 81.36 to 98.34 when using the SGD parameters as an optimizer, while f1-score values ranged from 56.67 to 97.37 for all CNNs. Table 2 shows that DenseNet and VGG19 had the best accuracy for the evaluation with 25 epochs, with an accuracy of 98.34 and a learning rate of respectively 0.001 and 0.01. Looking at the f1-score result the DenseNet (97.38%) had the best performance with 25 epochs. Furthermore, for 10 epochs DenseNet once again performed comparatively better with the highest accuracy (97.84) and f1-score (96.63) for a learning rate of 0.01. On the other hand, the lowest performance recorded for 10 and 25 epochs was SqueezeNet with an accuracy of respectively 81.23 and 81.36 and an f1-score of respectively 57.02% and 51.46% for a learning rate of 0.1. According to the specificity metrics, the majority of the

Table 2. Performance evaluation of six CNN architectures, changing the value of the epoch, batch size, and learning rate using Stochastic Gradient Descent as an optimizer.

model	batch	Epoch	Learning rate	Accuracy	precision	Recall	specificity	F1-score	Time
ALEXNET	16	10	0.1	94.59%	91.50%	92.53%	96.10%	92.01%	286
	32			94.24%	91.43%	92.12%	95.75%	91.77%	263
	64			94.25%	91.41%	91.91%	95.71%	91.66%	250
DENSENET	16			97.67%	96.14%	96.72%	98.33%	96.43%	515
	32			97.33%	95.86%	96.28%	98.04%	96.07%	461
	64			97.84%	96.60%	96.66%	98.37%	96.63%	428
EFFICIENTNET	16			95.29%	93.07%	92.96%	96.34%	93.01%	497
	32			95.97%	93.76%	93.79%	96.94%	93.77%	478
	64			95.84%	93.52%	93.65%	96.85%	93.58%	457
GOOGLENET	16			95.13%	92.43%	92.49%	96.43%	92.46%	382
	32			95.80%	93.38%	93.55%	96.89%	93.46%	335
	64			95.47%	93.03%	93.08%	96.69%	93.05%	301
SQUEEZENET	16			97.33%	95.48%	96.31%	98.15%	95.89%	330
	32			95.97%	94.07%	94.27%	96.99%	94.17%	313
	64			81.23%	50.59%	65.31%	83.26%	57.02%	266
VGG19	16			93.73%	91.07%	90.42%	95.31%	90.74%	768
	32			94.77%	91.74%	92.28%	96.13%	92.01%	625
	64			93.39%	90.71%	89.61%	95.03%	90.16%	611
ALEXNET	16		0.01	93.73%	91.93%	90.03%	95.14%	90.97%	287
	32			95.11%	92.51%	92.67%	96.29%	92.59%	265
	64			95.11%	93.15%	92.64%	96.24%	92.89%	256
DENSENET	16			97.33%	95.72%	96.40%	98.07%	96.06%	512
	32			97.16%	95.40%	96.05%	97.96%	95.72%	457
	64			97.33%	95.82%	96.13%	98.05%	95.97%	424
EFFICIENTNET	16			95.80%	93.62%	93.64%	96.79%	93.63%	501
	32			95.46%	93.64%	92.87%	96.41%	93.25%	478
	64			96.32%	94.41%	94.74%	97.19%	94.57%	457
GOOGLENET	16			95.12%	92.49%	92.49%	96.36%	92.49%	384
	32			94.60%	91.53%	91.96%	95.97%	91.74%	374
	64			95.12%	92.80%	92.49%	96.29%	92.64%	339
SQUEEZENET	16			96.82%	95.05%	95.64%	97.68%	95.34%	336
	32			96.82%	95.11%	95.51%	97.67%	95.31%	309
	64			96.31%	94.24%	94.96%	97.32%	94.60%	263
VGG19	16			86.48%	78.35%	90.89%	85.15%	84.16%	1314
	32			93.40%	90.52%	89.40%	94.99%	89.96%	628
	64			93.58%	90.51%	89.67%	95.11%	90.09%	619

Continued

ALEXNET	16		94.08%	90.80%	91.94%	95.79%	91.37%	286
	32		95.80%	93.38%	93.76%	96.86%	93.57%	264
	64		94.60%	91.46%	92.38%	96.11%	91.92%	261
DENSENET	16		95.98%	93.87%	93.73%	96.95%	93.80%	505
	32		95.63%	93.32%	93.50%	96.76%	93.41%	455
	64		89.47%	88.58%	81.19%	91.84%	84.72%	427
EFFICIENTNET	16		95.11%	92.79%	92.28%	96.20%	92.53%	504
	32		94.07%	91.39%	91.04%	95.38%	91.21%	480
	64		90.88%	86.68%	86.09%	92.84%	86.38%	460
GOOGLENET	16	0.001	94.25%	91.15%	91.55%	95.67%	91.35%	421
	32		92.13%	89.72%	86.99%	93.72%	88.33%	377
	64		88.88%	86.24%	81.68%	90.98%	83.90%	340
SQUEEZENET	16		96.66%	94.79%	95.37%	97.54%	95.08%	330
	32		95.46%	92.96%	93.47%	96.66%	93.21%	310
	64		94.25%	91.25%	91.91%	95.79%	91.58%	263
VGG19	16		93.23%	90.89%	89.01%	94.83%	89.94%	662
	32		92.34%	88.38%	88.59%	94.33%	88.48%	435
	64		92.62%	89.69%	88.47%	94.37%	89.08%	262
ALEXNET	16		95.80%	93.33%	94.31%	96.94%	93.82%	697
	32		95.80%	93.66%	93.98%	96.85%	93.82%	637
	64		94.59%	92.11%	92.36%	95.95%	92.23%	616
DENSENET	16		97.50%	96.09%	95.94%	98.12%	96.01%	3593
	32		98.00%	96.75%	97.17%	98.54%	96.96%	2043
	64		98.34%	97.18%	97.49%	98.81%	97.33%	2300
EFFICIENTNET	16		96.66%	95.05%	94.91%	97.42%	94.98%	1226
	32		96.83%	95.21%	95.36%	97.57%	95.28%	1178
	64		96.48%	94.62%	94.92%	97.34%	94.77%	1129
GOOGLENET	16	25	95.80%	93.23%	93.94%	96.96%	93.58%	938
	32	0.1	95.30%	92.43%	92.63%	96.53%	92.53%	802
	64		95.13%	92.16%	92.85%	96.51%	92.50%	731
SQUEEZENET	16		81.36%	50.88%	65.34%	83.53%	57.21%	876
	32		97.50%	95.84%	96.49%	98.22%	96.16%	824
	64		73.21%	42.41%	65.43%	76.39%	51.46%	701
VGG19	16		98.34%	97.18%	97.49%	98.81%	97.33%	3943
	32		94.44%	91.77%	90.85%	95.79%	91.31%	1183
	64		95.30%	92.65%	92.60%	96.47%	92.62%	729

Continued

ALEXNET	16	0.01	94.77%	92.06%	91.59%	95.95%	91.82%	713
	32		96.14%	94.18%	94.33%	97.10%	94.25%	652
	64		95.97%	93.92%	93.94%	96.96%	93.93%	628
DENSENET	16		95.63%	93.39%	93.29%	96.67%	93.34%	3593
	32		98.34%	97.22%	97.55%	98.80%	97.38%	2043
	64		97.84%	96.55%	96.66%	98.42%	96.60%	2300
EFFICIENTNET	16		96.48%	94.72%	94.53%	97.29%	94.62%	1243
	32		95.80%	93.60%	93.67%	96.79%	93.63%	1193
	64		95.99%	94.18%	93.97%	96.88%	94.07%	1142
GOOGLENET	16		95.46%	92.72%	93.32%	96.71%	93.02%	948
	32		94.95%	92.27%	92.34%	96.22%	92.30%	807
	64		94.42%	91.50%	91.57%	95.79%	91.53%	734
SQUEEZENET	16		97.33%	95.86%	96.37%	98.04%	96.11%	862
	32		97.50%	96.28%	96.52%	98.14%	96.40%	843
	64		97.17%	95.73%	96.05%	97.91%	95.89%	817
VGG19	16		94.09%	91.09%	90.77%	95.55%	90.93%	764
	32		93.58%	91.04%	89.91%	95.17%	90.47%	619
	64		94.36%	91.36%	91.08%	95.76%	91.22%	628
ALEXNET	16	0.001	95.11%	92.31%	93.15%	96.45%	92.73%	708
	32		95.46%	93.11%	93.41%	96.59%	93.26%	661
	64		94.08%	90.80%	91.94%	95.79%	91.37%	633
DENSENET	16		97.16%	95.32%	96.23%	97.98%	95.77%	1894
	32		96.66%	94.64%	95.32%	97.58%	94.98%	1850
	64		96.32%	94.56%	94.35%	97.22%	94.45%	2310
EFFICIENTNET	16		95.81%	93.43%	94.07%	96.87%	93.75%	1244
	32		95.46%	93.30%	93.11%	96.47%	93.20%	1196
	64		93.89%	91.55%	90.35%	95.18%	90.95%	1147
GOOGLENET	16		97.67%	96.04%	96.72%	98.36%	96.38%	976
	32		94.42%	92.37%	91.33%	95.67%	91.85%	848
	64		91.24%	88.97%	85.69%	92.97%	87.30%	768
SQUEEZENET	16		96.15%	94.04%	94.48%	97.19%	94.26%	5524
	32		96.66%	94.87%	95.28%	97.53%	95.07%	5564
	64		95.46%	93.13%	93.90%	96.68%	93.51%	3144
VGG19	16		93.57%	90.70%	89.82%	95.10%	90.26%	703
	32		93.05%	90.00%	89.08%	94.74%	89.54%	675
	64		92.52%	89.20%	88.67%	94.36%	88.93%	677

misclassifications are related to the SqueezeNet architecture (76.39%) for a batch size of 64, the learning rate of 0.1, and 25 epochs.

Table 3 compares the CNN architectures when utilizing the Adaptive Moment Estimation (Adam) parameter as an optimizer. The results indicated that DenseNet (98.34%) had the best accuracy for 10-epochs using a learning rate of 0.01 and a batch size of 16. While EfficientNet b7 (89.46%) and VGG19 (91.25%) had the lowest accuracy with a learning rate of 0.1. The highest f1-score percentages recorded for the Adam optimizer were obtained by SqueezeNet (97.65%) and DenseNet (97.34%) for a learning rate of 0.01. EfficientNet b7 had the worst f1-score performance 85.33%, with a learning rate of 0.1 and a batch size of 16.

Table 3. Performance evaluation of six CNN architectures, changing the value of the epoch, batch size, and learning rate using Adaptive Moment Estimation (Adam) as an optimizer.

model	batch	Epoch	Learning rate	accuracy	precision	recall	specificity	F1-score	Time
ALEXNET	16	10	0.1	93.03%	89.44%	90.50%	94.75%	89.97%	303
	32			94.61%	92.42%	91.96%	95.85%	92.19%	280
	64			93.89%	91.32%	90.11%	95.16%	90.71%	264
DENSENET	16			97.50%	96.09%	95.94%	98.12%	96.01%	446
	32			96.82%	94.93%	95.42%	97.70%	95.17%	431
	64			98.01%	96.73%	97.17%	98.55%	96.95%	411
EFFICEINTNET	16			89.46%	88.05%	82.78%	91.39%	85.33%	1561
	32			93.03%	89.61%	90.99%	94.87%	90.29%	1483
	64			92.31%	88.94%	89.49%	94.21%	89.21%	1452
GOOGLENET	16			94.59%	91.88%	91.60%	95.87%	91.74%	484
	32			94.95%	92.02%	92.25%	96.30%	92.13%	417
	64			94.77%	91.82%	93.01%	96.36%	92.41%	315
SQUEEZENET	16			96.49%	94.28%	94.86%	97.45%	94.57%	356
	32			95.30%	93.35%	91.73%	96.24%	92.53%	327
	64			95.29%	93.28%	92.30%	96.27%	92.79%	282
VGG19	16			91.25%	87.42%	87.60%	93.49%	87.51%	278
	32			94.24%	91.10%	91.79%	95.71%	91.44%	272
	64			94.39%	92.45%	90.47%	95.73%	91.45%	264
ALEXNET	16		0.01	92.87%	89.68%	89.87%	94.61%	89.77%	304
	32			92.52%	89.69%	89.61%	94.40%	89.65%	279
	64			92.66%	89.02%	90.30%	94.72%	89.66%	267
DENSENET	16			98.17%	96.98%	97.37%	98.67%	97.17%	756
	32			95.46%	93.64%	92.87%	96.41%	93.25%	712
	64			97.33%	95.74%	96.22%	98.05%	95.98%	672
EFFICEINTNET	16			94.08%	91.36%	91.10%	95.38%	91.23%	457
	32			95.46%	93.00%	93.68%	96.58%	93.34%	441
	64			94.59%	91.94%	92.29%	95.92%	92.11%	421

Continued

GOOGLENET	16			95.30%	92.54%	93.60%	96.70%	93.07%	400
	32			94.94%	91.98%	92.76%	96.34%	92.37%	349
	64			94.43%	91.21%	91.21%	95.79%	91.21%	316
SQUEEZENET	16			97.84%	96.34%	96.78%	98.46%	96.56%	354
	32			97.84%	96.23%	96.87%	98.51%	96.55%	332
	64			98.34%	97.16%	97.52%	98.82%	97.34%	282
VGG19	16			93.72%	91.17%	90.17%	95.05%	90.67%	280
	32			93.75%	90.77%	89.93%	95.26%	90.35%	266
	64			94.28%	91.70%	90.70%	95.67%	91.20%	260
ALEXNET	16			94.94%	91.99%	92.88%	96.34%	92.43%	298
	32			94.42%	91.69%	92.51%	95.92%	92.10%	265
	64			95.45%	93.15%	93.17%	96.53%	93.16%	249
DENSENET	16			98.17%	96.92%	97.34%	98.69%	97.13%	540
	32			98.34%	97.10%	97.70%	98.85%	97.40%	488
	64			98.01%	96.73%	97.01%	98.54%	96.87%	457
EFFICEINTNET	16			97.00%	95.64%	95.69%	97.66%	95.66%	463
	32			95.80%	93.57%	93.88%	96.81%	93.72%	438
	64		0.001	96.65%	95.05%	94.85%	97.41%	94.95%	419
GOOGLENET	16			95.12%	92.51%	92.21%	96.29%	92.36%	404
	32			95.30%	92.65%	92.72%	96.55%	92.68%	345
	64			95.47%	92.92%	92.99%	96.63%	92.95%	314
SQUEEZENET	16			97.50%	96.37%	96.28%	98.14%	96.32%	355
	32			96.65%	94.66%	95.37%	97.59%	95.01%	315
	64			96.82%	94.90%	95.70%	97.72%	95.30%	282
VGG19	16			95.64%	93.13%	93.19%	96.75%	93.16%	276
	32			95.63%	93.31%	93.44%	96.67%	93.37%	273
	64			95.64%	93.13%	93.19%	96.75%	93.16%	251
ALEXNET	16			92.52%	89.69%	89.37%	94.32%	89.53%	718
	32			94.59%	91.57%	92.68%	96.12%	92.12%	671
	64			94.08%	90.75%	91.79%	95.72%	91.27%	653
DENSENET	16			96.15%	93.82%	94.93%	97.33%	94.37%	3975
	32	25	0.1	97.50%	96.05%	96.46%	98.16%	96.25%	1183
	64			96.83%	94.91%	95.72%	97.68%	95.31%	1097
EFFICEINTNET	16			94.25%	91.46%	91.28%	95.54%	91.37%	1152
	32			94.94%	92.32%	92.64%	96.15%	92.48%	1457
	64			95.46%	93.38%	93.14%	96.47%	93.26%	1556

Continued

GOOGLENET	16	0.01	95.11%	92.59%	92.55%	96.26%	92.57%	1011
	32		94.09%	90.61%	90.68%	95.58%	90.64%	870
	64		93.73%	90.18%	90.99%	95.48%	90.58%	785
SQUEEZENET	16		96.70%	93.07%	95.27%	97.01%	94.16%	6522
	32		97.67%	96.28%	96.39%	98.26%	96.33%	6085
	64		97.16%	95.53%	95.74%	97.87%	95.63%	6344
VGG19	16		94.94%	92.21%	92.46%	96.23%	92.33%	694
	32		93.38%	90.92%	88.89%	94.80%	89.89%	662
	64		95.11%	92.56%	92.46%	96.29%	92.51%	651
ALEXNET	16		91.96%	87.95%	88.60%	94.02%	88.27%	696
	32		94.07%	92.11%	90.59%	95.31%	91.34%	639
	64		92.31%	89.94%	87.92%	93.92%	88.92%	600
DENSENET	16		98.17%	96.98%	97.37%	98.67%	97.17%	3943
	32		97.33%	95.76%	96.34%	98.05%	96.05%	3720
	64		98.01%	96.61%	97.29%	98.60%	96.95%	1107
EFFICEINTNET	16		96.82%	95.28%	95.30%	97.54%	95.29%	2163
	32		96.32%	94.65%	94.38%	97.13%	94.51%	1179
	64		95.80%	93.65%	94.01%	96.81%	93.83%	1122
GOOGLENET	16		96.15%	93.71%	94.60%	97.27%	94.15%	984
	32		94.08%	90.70%	91.13%	95.66%	90.91%	844
	64		95.29%	92.48%	93.54%	96.68%	93.01%	759
SQUEEZENET	16		98.25%	97.08%	97.60%	98.94%	97.34%	935
	32		97.67%	96.11%	96.57%	98.33%	96.34%	873
	64		96.83%	95.33%	95.58%	97.62%	95.45%	795
VGG19	16		93.48%	90.26%	90.28%	95.03%	90.27%	695
	32		94.61%	92.18%	91.11%	95.92%	91.64%	662
	64		93.58%	90.82%	89.30%	95.07%	90.05%	689
ALEXNET	16	0.001	96.14%	93.94%	94.08%	97.11%	94.01%	690
	32		95.80%	93.72%	93.49%	96.74%	93.60%	649
	64		95.28%	92.70%	93.39%	96.51%	93.04%	615
DENSENET	16		97.67%	96.04%	96.72%	98.36%	96.38%	3972
	32		98.00%	96.68%	97.29%	98.57%	96.98%	1940
	64		98.01%	96.64%	97.11%	98.58%	96.87%	1143
EFFICEINTNET	16		96.31%	95.02%	93.96%	97.06%	94.49%	1238
	32		95.98%	93.84%	93.88%	96.92%	93.86%	1180
	64		96.49%	94.63%	95.07%	97.33%	94.85%	1128

Continued

GOOGLENET	16	95.30%	92.63%	92.69%	96.56%	92.66%	989
	32	94.78%	91.78%	92.01%	96.15%	91.89%	845
	64	95.12%	92.34%	92.58%	96.42%	92.46%	762
SQUEEZENET	16	97.67%	96.47%	96.69%	98.28%	96.58%	872
	32	97.16%	95.48%	96.14%	97.95%	95.81%	809
	64	97.16%	95.62%	96.05%	97.90%	95.83%	702
VGG19	16	94.43%	92.13%	91.21%	95.77%	91.67%	691
	32	94.96%	92.29%	91.80%	96.23%	92.04%	677
	64	94.45%	92.00%	90.88%	95.83%	91.44%	661

Table 2 and Table 3 indicate that for most of the hyperparameters, all CNN models generated a somewhat greater percentage of recall than precision for both Adam and SGD optimizers. When compared to the training labels, a system with high recall but low precision returns many results, but the majority of its predicted labels are inaccurate. High scores for both, on the other hand, indicate that the classifier is producing accurate (high precision) results as well as a majority of all positive results (high recall). SqueezeNet had the worst performance in this evaluation, scoring more than a 10% difference between precision and recall during testing (precision = 42.41 and recall = 65.43 percent) with the SGD optimizer, a learning rate of 0.1, batch size of 64, and 2 epochs.

Table 2 and Table 3 show that across all hyperparameters for the time parameter, there was no clear pattern for selecting the architecture with the least training time. AlexNet's values tended to be low. The AlexNet model had the lowest values with 250 seconds in epochs (10), batch (64), and learning rate (0.1). On the other hand, AlexNet's accuracy and f1-score were frequently lower than those reported by other CNN architectures. With a learning rate of 0.1 and a batch size of 1, SqueezeNet (6522) had the slowest training speeds in 25 epochs (16). Finally, the EfficientNet b7 architecture required the most training time for the majority of hyperparameters. The longest training times had no bearing on the accuracy, regardless of how high or low it was.

5. Discussion

The optimizer plays a key role in minimizing the error function, allowing the model to conform to the instances in the training set. For both Adam and SGD optimizer, the validation accuracy and F1-score of CNNs in detecting early blight and late blight disease ranged from 81.23 to 98.51 and 56.67 to 97.37 respectively. It was observed that training the network with Adam as the optimizer increased the accuracy of all models to be between 89.46 and 98.51 as shown in Table 3. However, the GoogleNet architecture may have contributed to poor validation accuracy using the Adam optimizer since it starts with a large receptive field to decrease computing requirements [12]. These findings are consistent

with those of Russakovsky *et al.* After tuning the hyperparameters, it was also observed that the highest accuracy and f1-score for detection of potato late blight and early blight disease was obtained from DenseNet (97.37). The findings reveal that a larger batch size does not always result in high accuracy, and that the learning rate and optimizer utilized can have a major impact. Lowering the learning rate and batch size will help the network to train more efficiently, especially when fine-tuning. Our findings are consistent with those of Masters *et al.* [35], who suggested that smaller batch sizes be employed. Radiuk *et al.* [36] claim that when a high learning rate is utilized, the larger the batch size, the better the CNN's performance. While we do not encourage using big batch size numbers in our research, Radiuk's findings are consistent with our findings on the batch size and learning rate relationship. We specifically mentioned that better learning rates necessitate bigger batch sizes. Finally, Bengio *et al.* [37] suggested that a batch size of 32 is a good place to start. While our trials (which showed that a batch size of 32 produced decent results) back this up, the best results were obtained with a batch size of 16.

The lowest performance recorded for 10 and 25 epochs was SqueezeNet with an accuracy of respectively 73.21% and an f1-score of 50.86% with SGD optimizer for a learning rate of 0.1 and a batch size of 64. The best performance (98.34% validation accuracy and 97.37 f1-score) was obtained using a combination of batch (32), learning rate (0.01), and epochs (25).

Table 4 compares this study to similar studies in which the best-performing CNN designs have been reported. Yen Lee *et al.* [19] focused solely on constructing a CNN architecture for disease detection on potatoes, achieving a 92 percent accuracy. Similarly, Tiwari *et al.* [21] used fine-tuning (transfer learning) to extract significant characteristics from the dataset using pre-trained models like VGG19. The results were then analyzed using multiple classifiers, with logistic regression outperforming the others by a significant margin of classification accuracy, achieving 97.8% over the test dataset. Multiclass SVM was used by Islam *et al.* [15] to divide leaf images into three classes based on ten color and textural variables. The cross-validated accuracy of the system was 93.7 percent. Finally, this study evaluated the performance of six CNNs using transfer learning and modifying four hyperparameters, resulting in the DenseNet model with the

Table 4. Comparaison of the conducted study with similar studies.

Reference	Disease	Dataset size	Methodology	Accuracy
[18]	Early and late blight	1150	CNN	92%
[20]	Early and late blight	2152	SVM, KNN, and Neural NetCNN	97.8%
[14]	Early and late blight	300	Segment and Multi SVM	95%
This study	Early and late blight	4072	AlexNet, GoogleNet, EfficientNet_b7, SqueezeNet, VGG19. And DenseNet	98.34%

highest performance, with 98.34% validation accuracy and 97.37% f1-score. Similarly, VGG 16, Inception V4, ResNet with 50, 101, and 152 layers, and DenseNets with 121 layers were all tested by Too *et al.* [38]. The investigation employed data from PlantVillage from 38 different classes, including damaged and healthy photos of leaves from 14 different plants. DenseNets121 produced the best results, with a testing accuracy of 99.75 percent.

6. Conclusion

The agriculture industry can considerably benefit from the use of CNN in image classification to boost yield production. The performance of five CNN models (VGG19, AlexNet, GoogleNet, EfficientNet-b7, SqueezeNet, and DenseNet) was evaluated in this study. The epochs, batch, optimizer, and learning rate hyperparameters of all the CNN architectures were varied. The learning rate and batch size, according to our findings, have a substantial impact on the network's performance. The learning rate and batch size have a strong relationship; when learning rates are high, a big batch size performs better than when learning rates are low. The results showed that DenseNet architecture, which used the SGD optimizer, had the greatest performance for early and late blight disease diagnosis with 98.34% validation accuracy and 97.37% F1-score. SqueezeNet architecture had the lowest performance 73.21% accuracy and an f1-score of 50.86% when utilizing the SGD optimizer. Overall, the Adam optimizer outperforms the SGD optimizer, although the SGD optimizer yields the best results. The SGD optimizer has no visible effect on reaching high accuracy.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Ribeiro, P.F. and Rodriguez, A.V.C. (2020) Emerging Advanced Technologies to Mitigate the Impact of Climate Change in Africa. *Plants*, **9**, Article No. 381. https://doi.org/10.3390/plants9030381

[2] Devaux, A., Kromann, P. and Ortiz, O. (2014) Potatoes for Sustainable Global Food Security. *Potato Research*, **57**, 185-199. https://doi.org/10.1007/s11540-014-9265-1

[3] Olanya, O.M., Adipala, E., Hakiza, J.J., *et al.* (2001) Epidemiology and Population Dynamics of Phytophthora Infestans in Sub-Saharan Africa: Progress and Constraints. *African Crop Science Journal*, **9**, 185-193. https://doi.org/10.4314/acsj.v9i1.27638

[4] International Potato Center (2020). CIP in Uganda. International Potato Center, Lima, 4 p.

[5] Brancato, A., Brocca, D., Cabrera, L.C., *et al.* (2018) Reporting Data on Pesticide Residues in Food and Feed According to Regulation (EC) No 396/2005 (2017 Data Collection). *EFSA Journal*, **16**, e05285. https://doi.org/10.2903/j.efsa.2018.5285

[6] Bauske, M.J., Robinson, A.P. and Gudmestad, N.C. (2018) Early Blight in Potato. NDSU Extension.

https://www.ag.ndsu.edu/publications/crops/early-blight-in-potato

[7] Robinson, A. (2017) Late Blight in Potato. NDSU Extension. https://www.ndsu.edu/agriculture/

[8] Ayu, H.R., Surtono, A. and Apriyanto, D.K. (2021) Deep Learning for Detection Cassava Leaf Disease. *Journal of Physics*: *Conference Series*, **1751**, Article ID: 012072. https://doi.org/10.1088/1742-6596/1751/1/012072

[9] Aitelkadi, K., Bakouri, S., Belbrik, M., Hajji, H. and Chtaina, N. (2020) Expérimentation d'un modèle de détection précoce des maladies de la tomate par apprentissage profond [Experimentation of Model for Early Detection of Tomato Diseases by Deep Learning]. *Revue Marocaine de Protection des Plantes*, No. 14, 19-30.

[10] Rudakov, N., Eerola, T., Lensu, L., Kälviäinen, H. and Haario, H. (2019) Detection of Mechanical Damages in Sawn Timber Using Convolutional Neural Networks. In: Brox, T., Bruhn, A. and Fritz, M., Eds., *Pattern Recognition*. *GCPR* 2018. *Lecture Notes in Computer Science*, Springer, Cham, 115-126. https://doi.org/10.1007/978-3-030-12939-2_9

[11] Ashraf, S., Kadery, I., Chowdhury, A.A., Mahbub, T.Z. and Rahman, R.M. (2019) Fruit Image Classification Using Convolutional Neural Networks. *International Journal of Software Innovation*, **7**, 51-70. https://doi.org/10.4018/IJSI.2019100103

[12] KC, K., Yin, Z., Wu, M. and Wu, Z. (2019) Depthwise Separable Convolution Architectures for Plant Disease Classification. *Computers and Electronics in Agriculture*, **165**, Article ID: 104948. https://doi.org/10.1016/j.compag.2019.104948

[13] Fuentes, A., Yoon, S., Kim, S.C. and Park, D.S. (2017) A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition. *Sensors*, **17**, Article No. 2022. https://doi.org/10.3390/s17092022

[14] Mohanty, S.P., Hughes, D.P. and Salathé, M. (2016) Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, **7**, Article No. 1419. https://doi.org/10.3389/fpls.2016.01419

[15] Islam, M., Dinh, A., Wahid, K. and Bhowmik, P. (2017) Detection of Potato Diseases Using Image Segmentation and Multiclass Support Vector Machine. 2017 *IEEE* 30*th Canadian Conference on Electrical and Computer Engineering* (*CCECE*), Windsor, 30 April-3 May 2017, 1-4. https://doi.org/10.1109/CCECE.2017.7946594

[16] Hughes, D.P. and Salathe, M. (2015) An Open Access Repository of Images on Plant Health to Enable the Development of Mobile Disease Diagnostics. arXiv:1511.08060. http://arxiv.org/abs/1511.08060

[17] Sanjeev, K., Gupta, N.K., Jeberson, W.J. and Paswan, S. (2021) Early Prediction of Potato Leaf Diseases Using ANN Classifier. *Oriental Journal of Computer Science and Technology*, **13**, 129-134. https://doi.org/10.13005/ojcst13.0203.11

[18] Sharkawy, A.N. (2020) Principle of Neural Network and Its Main Types: Review. *Journal of Advances in Applied & Computational Mathematics*, **7**, 8-19. https://doi.org/10.15377/2409-5761.2020.07.2

[19] Lee, T.Y., Yu, J.Y., Chang, Y.C. and Yang, J.M. (2020) Health Detection for Potato Leaf with Convolutional Neural Network. *Proceedings of the* 2020 *Indo— Taiwan* 2*nd International Conference on Computing, Analytics and Networks* (*Indo-Taiwan ICAN*), Rajpura, 7-15 February 2020, 289-293. https://doi.org/10.1109/Indo-TaiwanICAN48429.2020.9181312

[20] Hossin, M and Sulaiman, M.N. (2015) A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, **5**, 11 p. https://doi.org/10.5121/ijdkp.2015.5201

[21] Tiwari, D., Ashish, M., Gangwar, N., Sharma, A., Patel, S. and Bhardwaj, S. (2020) Potato Leaf Diseases Detection Using Deep Learning. *Proceedings of the International Conference on Intelligent Computing and Control Systems* (*ICICCS* 2020), Hyderabad,, 25-26 June 202, 461-466. https://doi.org/10.1109/ICICCS48265.2020.9121067

[22] Chopra, K. and Srimathi, C. (2018) Logistic Regression and Convolutional Neural Networks Performance Analysis Based on Size of Dataset. *International Journal of Engineering Development and Research*, **6**, 292-295. https://www.ijedr.org/

[23] Oppenheim, D. and Shani, G. (2017) Potato Disease Classification Using Convolution Neural Networks. *Advances in Animal Biosciences*, **8**, 244-249. https://doi.org/10.1017/S2040470017001376

[24] Afzaal, H., Farooque, A.A., Schumann, A,W., *et al.* (2021) Detection of a Potato Disease (Early Blight) Using Artificial Intelligence. *Remote Sensing*, **13**, Article No. 411. https://doi.org/10.3390/rs13030411

[25] Rashid, J., Khan, I., Ali, G., Almotiri, S.H., Alghamdi, M.A. and Masood, K. (2021) Multi-Level Deep Learning Model for Potato Leaf Disease Recognition. *Electronics*, **10**, Article No. 2064. https://doi.org/10.3390/electronics10172064

[26] Ducher, J.F. and Esling, P. (2020) Folded CQT RCNN for Real-Time Recognition of Instrument Playing Techniques. International Society for Music Information Retrieval, Delft.

[27] Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2017) ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the ACM*, **60**, 84-90. https://dl.acm.org/doi/10.1145/3065386

[28] Szegedy, C. and Liu, W. (2015) Going Deeper with Convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Boston, 7-12 June 2015, 1-9. https://doi.org/10.1109/CVPR.2015.7298594

[29] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. (2015) Rethinking the Inception Architecture for Computer Vision. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, 27-30 June 2016, 2818-2826. http://arxiv.org/abs/1512.00567 https://doi.org/10.1109/CVPR.2016.308

[30] Szegedy, C., Ioffe, S., Vanhoucke, V. and Alemi, A.A. (2016) Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv:1602.07261. https://www.aaai.org/

[31] Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J. and Keutzer, K. (2016) SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and <0.5MB Model Size. arXiv:1602.07360. http://arxiv.org/abs/1602.07360

[32] Tan, M. and Le, Q.V. (2019) EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv:1905.11946.

[33] Simonyan, K. and Zisserman, A. (2014) Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556. http://arxiv.org/abs/1409.1556

[34] Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K.Q. (2017) Densely Connected Convolutional Networks. 2017 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, 21-26 July 2017, 4700-4708. https://doi.org/10.1109/CVPR.2017.243

[35] Masters, D. and Luschi, C. (2018) Revisiting Small Batch Training for Deep Neural Networks. arXiv:1804.07612. http://arxiv.org/abs/1804.07612

[36] Radiuk, P.M. (2018) Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets. *Information Technology and*

Management Science, **20**, 20-24. https://doi.org/10.1515/itms-2017-0003

[37] Bengio, Y. (2012) Practical Recommendations for Gradient-Based Training of Deep Architectures. In: Montavon, G., Orr, G.B. and Müller, K.R., Eds., *Neural Networks: Tricks of the Trade*, Springer, Berlin, 437-478. https://doi.org/10.1007/978-3-642-35289-8_26

[38] Too, E.C., Li, Y.J., Njuki, S. and Liu, Y.C. (2019) A Comparative Study of Fine-Tuning Deep Learning Models for Plant Disease Identification. *Computers and Electronics in Agriculture*, **161**, 272-279. https://doi.org/10.1016/j.compag.2018.03.032